```
   SSSSSSSSSSSSS   DDDDDDDDDDDD          AAAAAAAAAA
   SSSSSSSSSSSSS   DDDDDDDDDDDD          AAAAAAAAAA
   SSSSSSSSSSSSS   DDDDDDDDDDDD          AAAAAAAAAA
SSS                DDD          DDD  AAA           AAA
SSS                DDD          DDD  AAA           AAA
SSS                DDD          DDD  AAA           AAA
SSS                DDD          DDD  AAA           AAA
SSS                DDD          DDD  AAA           AAA
SSS                DDD          DDD  AAA           AAA
   SSSSSSSSSS      DDD          DDD  AAA           AAA
   SSSSSSSSSS      DDD          DDD  AAA           AAA
   SSSSSSSSSS      DDD          DDD  AAA           AAA
             SSS   DDD          DDD  AAAAAAAAAAAAAAAAA
             SSS   DDD          DDD  AAAAAAAAAAAAAAAAA
             SSS   DDD          DDD  AAAAAAAAAAAAAAAAA
             SSS   DDD          DDD  AAA           AAA
             SSS   DDD          DDD  AAA           AAA
             SSS   DDD          DDD  AAA           AAA
SSSSSSSSSSSSS      DDDDDDDDDDDD      AAA           AAA
SSSSSSSSSSSSS      DDDDDDDDDDDD      AAA           AAA
SSSSSSSSSSSSS      DDDDDDDDDDDD      AAA           AAA
```

```
_$

Sy
--
CT
CT
CT
CT
CU
CU
CU
DC
DC
DC
DC
DC
DC
DE
DE
DE
DE
DE
DE
DE
DE
DI
DI
DI
DI
DI
DI
DI
DI
DI
DI
DI
DI
DI
DU
DU
DU
DU
DU
DU
DU
DU
DV
DV
DV
DY
DY
DY
DY
DY
ER
ES
```

FILEID**QAST

```
  QQQQQQ      AAAAAA      SSSSSSSS  TTTTTTTTTT
  QQQQQQ      AAAAAA      SSSSSSSS  TTTTTTTTTT
QQ      QQ  AA      AA  SS              TT
QQ      QQ  AA      AA  SS              TT
QQ      QQ  AA      AA  SS              TT
QQ      QQ  AA      AA  SS              TT
QQ      QQ  AA      AA    SSSSSS        TT
QQ      QQ  AA      AA    SSSSSS        TT
QQ  QQ  QQ  AAAAAAAAAA          SS      TT
QQ  QQ  QQ  AAAAAAAAAA          SS      TT
QQ    QQ    AA      AA          SS      TT              ....
QQ    QQ    AA      AA          SS      TT              ....
  QQQQ  QQ  AA      AA  SSSSSSSS        TT              ....
  QQQQ  QQ  AA      AA  SSSSSSSS        TT              ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
QA
Sy

$$
AC
AC
AC
AC
AC
AC
CO
CO
DE
EX
EX
EX
EX
GE
IP
IP
MM
MM
PC
PC
PC
PC
PC
PH
PH
PH
PK
PK
PK
PK
PK
PK
PK
PK
PK
PK
PK
PK
PK
PK
PK
PR
PR
PS
PS
PS
PT
PT
QA
QA
QA
RE
RE
RP
RP
RP
```

Table of contents

```
0000     1          .TITLE  QAST - GET DATA FROM ANOTHER PROCESS
0000     2          .SBTTL  COPYRIGHT NOTICE
0000     3          .IDENT  'V04-000'
0000     4 ;
0000     5 ;******************************************************************************
0000     6 ;*                                                                            *
0000     7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                   *
0000     8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
0000     9 ;*  ALL RIGHTS RESERVED.                                                      *
0000    10 ;*                                                                            *
0000    11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
0000    12 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
0000    13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
0000    14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
0000    15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
0000    16 ;*  TRANSFERRED.                                                              *
0000    17 ;*                                                                            *
0000    18 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
0000    19 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0000    20 ;*  CORPORATION.                                                              *
0000    21 ;*                                                                            *
0000    22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
0000    23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
0000    24 ;*                                                                            *
0000    25 ;*                                                                            *
0000    26 ;******************************************************************************
0000    27 ;
```

```
0000     29          .SBTTL  PROGRAM DESCRIPTION
0000     30 ;++
0000     31 ;   FACILITY
0000     32 ;
0000     33 ;       SYSTEM DUMP ANALYZER
0000     34 ;
0000     35 ;   ABSTRACT
0000     36 ;
0000     37 ;       ROUTINES TO OBTAIN MEMORY FROM ANOTHER PROCESS ON
0000     38 ;       THE RUNNING SYSTEM.
0000     39 ;
0000     40 ;   ENVIRONMENT
0000     41 ;
0000     42 ;       NATIVE MODE, USER MODE
0000     43 ;
0000     44 ;   AUTHOR
0000     45 ;
0000     46 ;       TIM HALVORSEN, JULY 1978
0000     47 ;
0000     48 ;   MODIFIED BY
0000     49 ;
0000     50 ;       V03-003 MSH0011         Michael S. Harvey       23-Feb-1983
0000     51 ;               Simplify the handling of a target process in SUSP or SUSPO
0000     52 ;               state. Instead of having to queue yet another special kernel
0000     53 ;               AST just to resuspend the process, simply clear the RESPEN
0000     54 ;               bit (set by SDA's $RESUME call). This all works because now
0000     55 ;               the SUSPEND AST in the Exec is a normal kernel AST instead
0000     56 ;               of a special kernel AST and so SDA doesn't have to work as
0000     57 ;               hard as it used to in the case of a suspended process.
0000     58 ;
0000     59 ;               Use IPL$_SYNCH to close window between state test and special
0000     60 ;               kernel AST queueing.
0000     61 ;
0000     62 ;       V03-002 TMH0002         Tim Halvorsen           02-Aug-1983
0000     63 ;               Fix code which allows analysis of suspended processes
0000     64 ;               which was broken when EPIDs were added.
0000     65 ;
0000     66 ;       V03-001 KDM0002         Kathleen D. Morse       28-Jun-1982
0000     67 ;               Added $PRDEF.
0000     68 ;
0000     69 ;--
```

```
                                                I  8
                     0000     71          .SBTTL  DECLARATIONS
                     0000     72 ;
                     0000     73 ;        SYMBOL DEFINTIONS
                     0000     74 ;
                     0000     75
                     0000     76          $ACBDEF                         ; AST CONTROL BLOCK DEFINITIONS
                     0000     77          $PRIDEF                         ; PRIORITY DEFINITIONS
                     0000     78          $IPLDEF                         ; IPL DEFINITIONS
                     0000     79          $PCBDEF                         ; PROCESS CONTROL BLOCK
                     0000     80          $PHDDEF                         ; PROCESS HEADER
                     0000     81          $PRDEF                          ; PROCESSOR REGISTERS
                     0000     82          $PSLDEF                         ; PSL DEFINITIONS
                     0000     83          $SSDEF                          ; STATUS DEFINITIONS
                     0000     84          $STATEDEF                       ; PROCESS STATE VALUES
                     0000     85          $MCHKDEF                        ; MACHINE CHECK PROTECTION MASK
                     0000     86          $VADEF                          ; VIRTUAL ADDRESS DEFINITIONS
                     0000     87          $PTEDEF                         ; PAGE TABLE ENTRY DEFINITIONS
                     0000     88          $RPBDEF                         ; RESTART PARAMETER BLOCK
                     0000     89
                     0000     90 ;
                     0000     91 ;        DEFINE EXTENSIONS TO THE AST CONTROL BLOCK
                     0000     92 ;
                     0000     93          $DEFINI PKT
                     0000     94
 00000010            0000     95 PKT_ORIGPID      =       ACB$L_AST       ; PID OF REQUESTOR
 00000014            0000     96 PKT_ADDR         =       ACB$L_ASTPRM    ; ADDRESS OF REQUESTED DATA
                     0000     97
 0000001C            0000     98 .                =       ACB$L_KAST+4
                     001C     99
                     001C    100 $DEF     PKT_QAST        .BLKL           ; ADDRESS OF SCH$QAST
                     0020    101 $DEF     PKT_DEANON      .BLKL           ; ADDRESS OF EXE$DEANONPAGED
                     0024    102 $DEF     PKT_RETLOC      .BLKL           ; ADDRESS TO RETURN DATA
                     0028    103 $DEF     PKT_LEN         .BLKL           ; LENGTH OF DATA
                     002C    104 $DEF     PKT_STATUS      .BLKL           ; STATUS OF TRANSFER
                     0030    105 $DEF     PKT_STATLOC     .BLKL           ; ADDRESS TO RETURN STATUS
                     0034    106 $DEF     PKT_WAKE        .BLKL           ; ADDRESS OF SCH$WAKE
                     0038    107 ;$DEF    PKT_PRTCT       .BLKL           ; ADDRESS OF EXE$MCHK_PRTCT
                     0038    108 $DEF     PKT_IMGCNT      .BLKL           ; PHD$L_IMGCNT OF REQUESTOR
                     003C    109 $DEF     PKT_FLAGS       .BLKB           ; FLAGS BYTE
                     003D    110          _VIELD  PKT,0,<-
                     003D    111                  <SUSPEND,,M>,-          ; RE-SUSPEND PROCESS AFTER MEMORY FETCH
                     003D    112                  >
                     003D    113 $DEF     PKT_SIZE                        ; TOTAL SIZE OF FIXED PORTION
                     003D    114 $DEF     PKT_DATABUF                     ; START OF DATA TO BE MOVED
                     003D    115                                          ; AST CODE FOLLOWS DATA
                     003D    116          $DEFEND PKT
                     0000    117
                     0000    118          .DEFAULT DISPLACEMENT,LONG
                     0000    119
            00000000 0000    120          .PSECT  DATA,NOEXE,WRT
                     0000    121
                     0000    122 QAST_COUNT:
 00000000            0000    123          .LONG   0                       ; QAST REQUEST COUNTER
                     0004    124
            00000000 0000    125          .PSECT  QAST,EXE,NOWRT
                     0000    126
 00989680            0000    127 SECONDS =        10*1000*1000            ; 1 SECOND IN DELTA TIME
```

```
                        0000   128
                        0000   129 TIMEOUT:
FFFFFFFF FE363C80       0000   130         .LONG   -3*SECONDS,-1           ; 3 SECOND TIMEOUT COUNT
```

```
                                0008   132          .SBTTL  GETPROCMEM - GET MEMORY FROM ANOTHER PROCESS
                                0008   133 ;---
                                0008   134 ;
                                0008   135 ;       READ MEMORY FROM ANOTHER PROCESS ON THE RUNNING SYSTEM.
                                0008   136 ;
                                0008   137 ; INPUTS:
                                0008   138 ;
                                0008   139 ;       04(AP) = LOCATION TO READ IN OTHER PROCESS CONTEXT
                                0008   140 ;       08(AP) = ADDRESS OF BUFFER IN LOCAL MEMORY TO RECIEVE TRANSFER
                                0008   141 ;       12(AP) = LENGTH OF TRANSFER
                                0008   142 ;       16(AP) = PID OF OTHER PROCESS
                                0008   143 ;
                                0008   144 ; OUTPUTS:
                                0008   145 ;
                                0008   146 ;       SS$_NORMAL      - MEMORY TRANSFERRED OK
                                0008   147 ;       SS$_ACCVIO      - UNABLE TO ACCESS MEMORY
                                0008   148 ;       SS$_NOPRIV      - NOT ENOUGH PRIVILEGE (CMKRNL)
                                0008   149 ;       SS$_NONEXPR     - NON-EXISTANT PROCESS OR INVALID PID
                                0008   150 ;---
                        0000    0008   151          .ENTRY  GETPROCMEM,0
                                000A   152
0000022C 8F                DD   000A   153          PUSHL   #SS$_TIMEOUT            ; PRESET TO TIMED OUT STATUS
         5E                DD   0010   154          PUSHL   SP                      ; ADDRESS OF LONGWORD TO GET STATUS
      7E   0C AC           7D   0012   155          MOVQ    12(AP),-(SP)            ; MOVE LENGTH AND PID
      7E   04 AC           7D   0016   156          MOVQ    4(AP),-(SP)             ; MOVE SOURCE AND DESTINATION ADDRESSES
         05                DD   001A   157          PUSHL   #5                      ; NUMBER OF ARGUMENTS
         5E                DD   001C   158          PUSHL   SP                      ; ADDRESS OF ARGUMENT LIST
00000000'EF                9F   001E   159          PUSHAB  QAST                    ; ADDRESS OF KERNEL MODE ROUTINE
00000000'GF                02   FB   0024   160          CALLS   #2,G^SYS$CMKRNL         ; CALL ROUTINE IN KERNEL MODE
         48 50             E9   002B   161          BLBC    R0,90$                  ; IF ERROR, EXIT WITH STATUS
      5E   18              C0   002E   162          ADDL    #6*4,SP                 ; REMOVE ARGUMENT LIST
      03   50              D1   0031   163          CMPL    R0,#3                   ; AST OUTSTANDING?
           43              13   0034   164          BEQL    90$                     ; IF NOT, EXIT WITH SUCCESS
00000000'EF                D6   0036   165          INCL    QAST_COUNT              ; INCREMENT THE COUNTER
                                003C   166          $SETIMR_S ASTADR=B^QAST_TIMEOUT,-   ; SCHEDULE TIMEOUT REQUEST
                                003C   167                    REQIDT=QAST_COUNT,-       ; ASTPRM OF QAST COUNTER
                                003C   168                    DAYTIM=TIMEOUT            ; ADDRESS OF TIMEOUT DELTA TIME
                                0051   169          $HIBER_S                        ; WAIT FOR AST TO COMPLETE
                                0058   170          $CANTIM_S                       ; CANCEL OUTSTANDING TIMER REQUESTS
                                0061   171 ;
                                0061   172 ; HIBERNATE COULD HAVE COMPLETED DUE TO THE FOLLOWING REASONS:
                                0061   173 ;       1) WAKE FROM AST RESPONSE, REQUEST SUCCESSFUL
                                0061   174 ;       2) WAKE FROM TIMEOUT, REQUEST UNSUCCESSFUL, AST CANCELED BY TIMEOUT
                                0061   175 ;       3) WAKE FROM 'WAKE PENDING' FLAG, WHICH COMPLETES $HIBER IMMEDIATELY.
                                0061   176 ;               (NOT SURE WHAT SCENIOS CAUSE THIS, BUT BETTER SAFE...)
                                0061   177 ;
                                0061   178 ;       FOR CASE #3, WE CAN LIMIT THE DAMAGE BY CANCELING THE OUTSTANDING
                                0061   179 ;       AST SO THAT IT DOESN'T COME BACK AND WIPE OUT OUR STACK WITH THE
                                0061   180 ;       COMPLETION STATUS OR WIPE OUT OUR BUFFER WITH THE MEMORY.
                                0061   181 ;
0000022C 8F      6E            D1   0061   182          CMPL    (SP),#SS$_TIMEOUT       ; HAS REQUEST HAS COME BACK YET?
         0C      12            12   0068   183          BNEQ    50$                     ; BRANCH IF IT HAS
                                006A   184          $CMKRNL_S B^REJECT_RESPONSE     ; DONT LET AST EVER COME BACK
      50 8E      D0            D0   0076   185 50$:    MOVL    (SP)+,R0                ; GET RETURN STATUS
                 04            04   0079   186 90$:    RET                             ; EXIT WITH SUCCESS
```

```
                                007A   188          .SBTTL  QAST_TIMEOUT - AST ROUTINE CALLED WHEN QAST TIMES OUT
                                007A   189 ;---
                                007A   190 ;
                                007A   191 ;       THIS AST ROUTINE IS CALLED WHEN A SPECIAL KERNEL MODE
                                007A   192 ;       AST REQUEST TO ANOTHER PROCESS TIMES OUT.  THE IMAGE
                                007A   193 ;       COUNTER IS INCREMENTED CAUSING THE KAST ROUTINE (WHEN
                                007A   194 ;       IT EVER GETS GOING AGAIN) TO DROP IT ON THE FLOOR.  THE
                                007A   195 ;       CURRENT PROCESS IS WOKEN UP.  THE STATUS LONGWORD HAS
                                007A   196 ;       BEEN PRESET TO SS$_TIMEOUT SO THAT IT KNOWS THE REQUEST
                                007A   197 ;       FAILED.
                                007A   198 ;
                                007A   199 ; INPUTS:
                                007A   200 ;
                                007A   201 ;       4(AP) = QAST REQUEST NUMBER
                                007A   202 ;
                                007A   203 ; OUTPUTS:
                                007A   204 ;
                                007A   205 ;       NONE
                                007A   206 ;---
                                007A   207
                                007A   208 QAST_TIMEOUT:
                          0000  007A   209          .WORD   0
                                007C   210
00000000'EF     04 AC     D1   007C   211          CMPL    4(AP),QAST_COUNT        ; IS THIS FOR THE CURRENT QAST?
                    17    12   0084   212          BNEQ    90$                     ; IF NOT, IGNORE THE TIMEOUT
                                0086   213          $CMKRNL_S B^REJECT_RESPONSE     ; INCREMENT THE IMAGE COUNTER
                                0092   214          $WAKE_S                         ; WAKEUP THE CURRENT PROCESS
                          04    009D   215 90$:     RET
                                009E   216
                                009E   217 REJECT_RESPONSE:
                          0000  009E   218          .WORD   0
50      00000000'FF       D0   00A0   219          MOVL    @SCH$GL_CURPCB,R0       ; ADDRESS OF CURRENT PCB
     51      6C A0        D0   00A7   220          MOVL    PCB$L_PHD(R0),R1        ; ADDRESS OF PHD
          00F4 C1         D6   00AB   221          INCL    PHD$L_IMGCNT(R1)        ; INCREMENT IMAGE COUNTER
                          04   00AF   222          RET
```

```
                     00B0    224          .SBTTL  QAST - QUEUE MEMORY REQUEST TO ANOTHER PROCESS
                     00B0    225 ;---
                     00B0    226 ;
                     00B0    227 ;       QAST - QUEUE AST TO READ MEMORY FROM ANOTHER PROCESS
                     00B0    228 ;
                     00B0    229 ; INPUTS:
                     00B0    230 ;       04(AP) - LOCATION OF DATA
                     00B0    231 ;       08(AP) - RETURN LOCATION
                     00B0    232 ;       12(AP) - LENGTH OF TRANSFER
                     00B0    233 ;       16(AP) - PID OF TARGET PROCESS
                     00B0    234 ;       20(AP) - ADDRESS TO RETURN STATUS
                     00B0    235 ;
                     00B0    236 ; IMPLICIT INPUTS:
                     00B0    237 ;
                     00B0    238 ;       THE FOLLOWING SYMBOLS REFER TO LONGWORDS WHICH CONTAIN THE
                     00B0    239 ;       VALUE OF THE SYMBOL FOR THE CURRENT RUNNING EXECUTIVE:
                     00B0    240 ;
                     00B0    241 ;       SCH$GL_CURPCB
                     00B0    242 ;       SCH$GL_MAXPIX
                     00B0    243 ;       SCH$GL_PCBVEC
                     00B0    244 ;       PHV$GL_PIXBAS
                     00B0    245 ;       SGN$GL_BALSETCT
                     00B0    246 ;       SWP$GL_BALBASE
                     00B0    247 ;       SWP$GL_BSLOTSZ
                     00B0    248 ;       MMG$GL_SPTBASE
                     00B0    249 ;       EXE$GL_RPB
                     00B0    250 ;       EXE$AL[OCBUF
                     00B0    251 ;       EXE$DEANONPAGED
                     00B0    252 ;       EXE$MCHK_PRTCT
                     00B0    253 ;       SCH$QAST
                     00B0    254 ;       SCH$WAKE
                     00B0    255 ;
                     00B0    256 ; OUTPUTS:
                     00B0    257 ;
                     00B0    258 ;       R0 = 1 IF THE SPECIAL KERNEL MODE AST IS STILL OUTSTANDING
                     00B0    259 ;              (IMPLIES HIBERNATE NEEDED IN CALLING ROUTINE)
                     00B0    260 ;       R0 = 3 IF NO SPECIAL KERNEL AST WAS ISSUED (AVOID HIBERNATE)
                     00B0    261 ;
                     00B0    262 ;       SS$_ACCVIO = NO READ ACCESS TO MEMORY
                     00B0    263 ;       SS$_NONEXPR = NON-EXISTANT PROCESS OR INVALID PID
                     00B0    264 ;
                     00B0    265 ;---
                     00B0    266
                 007C 00B0    267 QAST:   .WORD   ^M<R2,R3,R4,R5,R6>
                     00B2    268
                     00B2    269 ;
                     00B2    270 ; CHECK ACCESSIBILITY OF SYSTEM VA BECAUSE ALTHOUGH A PROBE INSTRUCTION
                     00B2    271 ; WILL RETURN SUCCESS (PTE VALID), PAGEFAULT DOES NOT ALLOW ONE TO
                     00B2    272 ; FAULT IN SOME ONE ELSES PROCESS PAGE TABLE PAGE (WHOSE WORKING SET
                     00B2    273 ; DO YOU PUT IT IN?, ETC.) AND FAKES AN ACCESS VIOLATION ON THE MOVC.
                     00B2    274 ; THUS, WE MUST MUCK IN SYSTEM SPACE IN THE CONTEXT OF THE PROCESS WHICH
                     00B2    275 ; OWNS THE BALANCE SET SLOT TO AVOID PROBLEMS DISPLAYING HIS PROCESS
                     00B2    276 ; PAGE TABLE.  ALSO, CHECK IF BEYOND END OF SYSTEM VIRTUAL MEMORY, AS
                     00B2    277 ; PROBE DOES NOT DETECT THIS CONDITION, AND PAGEFAULT ABORTS ON IT.
                     00B2    278 ;
   56   10 AC   D0   00B2    279          MOVL    16(AP),R6                   ; ASSUME SWITCHING TO "CURRENT" PROCESS
55 04 AC    1F   E1   00B6    280          BBC     #VA$V_SYSTEM,4(AP),5$       ; CONTEXT SWITCH IF NOT SYSTEM SPACE
```

```
52   04 AC   000001FF 8F   CB  00BB           BICL3   #^X1FF,4(AP),R2           ; CLEAR PAGE OFFSET
50   52      00000000'FF   C3  00C4           SUBL3   @SWP$GL_BALBASE,R2,R0     ; BELOW BALANCE SET SLOTS?
                      45   19  00CC           BLSS    20$                       ; IF NOT, CHECK IF I/O SPACE
  50   50     F7      8F   78  00CE           ASHL    #-9,R0,R0                 ; COMPUTE BALANCE SET PAGE NUMBER
  50      00000000'FF      C6  00D3           DIVL    @SWP$GL_BSLOTSZ,R0        ; PROCESS HEADER INDEX
  00000000'FF         50   B1  00DA           CMPW    R0,@SGN$GL_BALSETCT       ; BEYOND END OF BALANCE SET SLOTS?
                      6C   18  00E1           BGEQ    8$                        ; IF SO, CHECK IF LEGAL AT ALL
  54      00000000'FF      D0  00E3           MOVL    @SCH$GL_CURPCB,R4         ; GET ADDRESS OF CURRENT PCB
          55    6C    A4   D0  00EA           MOVL    PCB$L_PHD(R4),R5          ; GET ADDRESS OF CURRENT PHD
          42 A5       50   B1  00EE           CMPW    R0,PHD$W_PHVINDEX(R5)     ; THIS PROCESS'S HEADER?
                      1C   13  00F2           BEQL    5$                        ; IF SO, OK TO USE THIS PROCESS CONTEXT
  51      00000000'FF      D0  00F4           MOVL    @PHV$GL_PIXBAS,R1         ; GET ADDRESS OF HEADER/PIX ARRAY
          50    6140       32  00FB           CVTWL   (R1)[R0],R0               ; GET PROCESS INDEX OWNING BALANCE SLOT
                      57   19  00FF           BLSS    80$                       ; ACCVIO IF NOBODY OWNS THE SLOT
  51      00000000'FF      D0  0101           MOVL    @SCH$GL_PCBVEC,R1         ; GET ADDRESS OF PCB ADDRESS ARRAY
          54    6140       D0  0108           MOVL    (R1)[R0],R4               ; GET PCB OWNING BALANCE SET SLOT
          56    60    A4   D0  010C           MOVL    PCB$L_PID(R4),R6          ; GET PID OF PROCESS WHICH OWNS SLOT
                    0073   31  0110      5$:  BRW     50$                       ; USE THAT PROCESS CONTEXT
                               0113      ;
                               0113      ;       SYSTEM ADDRESS IS BELOW THE BALANCE SET SLOTS.  CHECK IF MAPPED
                               0113      ;       BY ANY ACTIVE MEMORY CONTROLLER.  IF NOT, THEN ASSUME ITS I/O
                               0113      ;       SPACE AND DISALLOW TRANSFER.
                               0113      ;
  53   52   15   09        EF  0113     20$:  EXTZV   #VA$V_VPN,#VA$S_VPN,R2,R3 ; EXTRACT SYSTEM PAGE NUMBER
  51      00000000'FF      D0  0118           MOVL    @MMG$GL_SPTBASE,R1        ; GET VIRTUAL BASE OF SPT
          53    6143       D0  011F           MOVL    (R1)[R3],R3               ; GET PAGE TABLE ENTRY
                      37   18  0123           BGEQ    40$                       ; IF NOT VALID, CAN'T BE I/O SPACE
                               0125                                             ; (ALLOW TRANSFER TO CAUSE PAGEFAULT)
  53   53   15   00        EF  0125           EXTZV   #PTE$V_PFN,#PTE$S_PFN,R3,R3 ; GET PFN
  51      00000000'FF      D0  012A           MOVL    @EXE$GL_RPB,R1            ; GET RPB ADDRESS
       51      00BC   C1   9E  0131           MOVAB   RPB$L_MEMDSC(R1),R1       ; START OF MEMORY CONTROLLER DESC'S
               52     08   D0  0136           MOVL    #RPB$C_NMEMDSC,R2         ; SIZE OF ARRAY
                               0139           ASSUME  RPB$V_BASEPFN EQ 32
                               0139           ASSUME  RPB$S_BASEPFN EQ 32
   50    53    04     A1   C3  0139     25$:  SUBL3   4(R1),R3,R0               ; PFN LESS THAN BASE ADDRESS?
                      07   1F  013E           BLSSU   28$                       ; IF SO, SKIP TO NEXT ONE
  50    61    18      00   ED  0140           CMPZV   #RPB$V_PAGCNT,#RPB$S_PAGCNT,(R1),R0 ; WITHIN RANGE OF MEMORY?
                      15   1A  0145           BGTRU   40$                       ; IF SO, ALLOW ACCESS TO LOCATION
              51      08   C0  0147     28$:  ADDL    #RPB$C_MEMDSCSIZ,R1       ; SKIP TO NEXT DESCRIPTOR
              EC      52   F5  014A           SOBGTR  R2,25$                    ; AND LOOP UNTIL END OF ARRAY
                      09   11  014D           BRB     80$                       ; NOT MAPPED BY ANY CONTROLLER, ACCVIO
                               014F      ;
                               014F      ;       SYSTEM ADDRESS IS ABOVE THE BALANCE SET SLOTS.  CHECK IF BEYOND
                               014F      ;       END OF SYSTEM VIRTUAL ADDRESS SPACE.
                               014F      ;
  00000000'FF         52   D1  014F      8$:  CMPL    R2,@MMG$GL_MAXGPTE        ; LEGAL SYSTEM VA?
                      04   1F  0156           BLSSU   40$                       ; IF GPTE, DON'T NEED TO SWITCH CONTEXT
              50      0C   3C  0158     80$:  MOVZWL  #SS$_ACCVIO,R0            ; IF NOT, ACCESS VIOLATION
                           04  015B           RET
                               015C      ;
                               015C      ;       READ MEMORY FROM CURRENT PROCESS CONTEXT
                               015C      ;
                      7E   DC  015C     40$:  MOVPSL  -(SP)                     ; GET CURRENT PSL
  6E    02    16      00   F0  015E           INSV    #PSL$C_KERNEL,#PSL$V_PRVMOD,#PSL$S_PRVMOD,(SP)
          0000016A'EF      9F  0163           PUSHAB  42$                       ; ADDRESS FOLLOWING REI
                           02  0169           REI                               ; SET PREVIOUS MODE TO KERNEL
          52    04    AC   D0  016A     42$:  MOVL    4(AP),R2                  ; GET SOURCE BUFFER ADDRESS
```

```
                53   08 AC    7D  016E   338          MOVQ    8(AP),R3                ; GET DESTINATION ADDRESS AND LENGTH
                                  0172   339          IFNORD  R4,(R2),80$              ; CHECK FOR READ ACCESS
                                  0178   340          IFNOWRT R4,(R3),80$              ; CHECK FOR WRITE ACCESS
                                  017E   341 ;        PUSHAB  B^45$                    ; END OF RECOVERY BLOCK ADDESS
                                  017E   342 ;        MOVL    #<MCHK$M_LOG!MCHK$M_MCK!MCHK$M_NEXM!MCHK$M_UBA>,R0 ; PROTECT MASK
                                  017E   343 ;        JSB     @EXE$MCHK_PRTCT          ; INHIBIT MACHINE CHECKS
              63   62   54   28  017E   344          MOVC    R4,(R2),(R3)             ; MOVE DATA TO BUFFER
                                  0182   345 ;        RSB                              ; END OF PROTECTED CODE
                                  0182   346 ;45$:    BLBC    R0,80$                   ; BRANCH IF MACHINE CHECK OCCURRED
                    50   03   D0  0182   347          MOVL    #3,R0                    ; SET NO AST OUTSTANDING
                              04  0185   348 90$:     RET
                                  0186   349 ;
                                  0186   350 ;        READ MEMORY FROM SOME OTHER PROCESS CONTEXT
                                  0186   351 ;
                         56   D5  0186   352 50$:     TSTL    R6                       ; ANY PID TO SWITCH TO?
                         D2   13  0188   353          BEQL    40$                      ; BRANCH IF NOT
                   50    08E8 8F   3C  018A   354          MOVZWL  #SS$_NONEXPR,R0          ; ASSUME BAD PID
          51   00000000'FF    56   A3  018F   355          SUBW3   R6,@SCH$GL_MAXPIX,R1     ; CHECK FOR LEGAL INDEX
                         51   B6  0197   356          INCW    R1                       ; MAXPIX+1 = "SYSTEM PROCESS"
                         C1   13  0199   357          BEQL    40$                      ; SKIP AST IF "SYSTEM PROCESS"
                         E8   19  019B   358          BLSS    90$                      ; BR IF ILLEGAL INDEX
      51   0C AC    000000B1'8F   C1  019D   359          ADDL3   #PKT_SIZE+CODELEN,12(AP),R1 ; TOTAL SIZE OF BUFFER
                   00000000'FF   16  01A6   360          JSB     @EXE$ALLOCBUF            ; ALLOCATE BUFFER FOR CODE
                            D6   50  E9  01AC   361          BLBC    R0,90$                   ; BRANCH IF ERROR DETECTED
                         55   52   D0  01AF   362          MOVL    R2,R5                    ; SAVE ADDRESS OF PACKET
                   0C A5      56   D0  01B2   363          MOVL    R6,ACB$L_PID(R5)         ; SET TARGET PID
                0B A5    80 8F   90  01B6   364          MOVB    #1@ACB$V_KAST,ACB$B_RMOD(R5) ; SET SPECIAL KERNEL AST
                   50    0C AC   D0  01BB   365          MOVL    12(AP),R0                ; GET LENGTH OF TRANSFER
             18 A5    3D A540   9E  01BF   366          MOVAB   PKT_SIZE(R5)[R0],ACB$L_KAST(R5) ; SET ADDRESS FOR AST
                28 A5      50   D0  01C5   367          MOVL    R0,PKT_LEN(R5)           ; SET LENGTH OF TRANSFER
                14 A5    04 AC   D0  01C9   368          MOVL    4(AP),PKT_ADDR(R5)       ; SET ADDRESS FOR FETCH
                24 A5    08 AC   D0  01CE   369          MOVL    8(AP),PKT_RETLOC(R5)     ; AND ADDRESS OF RETURN LOCATION
                   2C A5    0C   D0  01D3   370          MOVL    #SS$_ACCVIO,PKT_STATUS(R5) ; ASSUME NO READ ACCESS
                30 A5    14 AC   D0  01D7   371          MOVL    20(AP),PKT_STATLOC(R5)   ; ADDRESS TO RETURN STATUS
          54    00000000'FF    D0  01DC   372          MOVL    @SCH$GL_CURPCB,R4        ; GET ADDRESS OF CURRENT PCB
                   50    6C A4   D0  01E3   373          MOVL    PCB$L_PHD(R4),R0         ; GET PHD ADDRESS
                10 A5    60 A4   D0  01E7   374          MOVL    PCB$L_PID(R4),PKT_ORIGPID(R5)    ; SET PID FOR RETURN
                38 A5    00F4 C0   D0  01EC   375          MOVL    PHD$L_IMGCNT(R0),PKT_IMGCNT(R5) ; SET IMGCNT OF REQUESTOR
                            3C A5   94  01F2   376          CLRB    PKT_FLAGS(R5)            ; AND CLEAR FLAGS BYTE
                         55   DD  01F5   377          PUSHL   R5                       ; SAVE REGS FOR MOVC
18 B5    0268'CF    0074'8F   28  01F7   378          MOVC3   #CODELEN,W^CODE,@ACB$L_KAST(R5) ; COPY CODE SEGMENT
                         55 8ED0  0200   379          POPL    R5                       ; RESTORE REGISTERS
       1C A5    00000000'EF    D0  0203   380          MOVL    SCH$QAST,PKT_QAST(R5)    ; COPY ABSOLUTE ADDRESSES IN EXECUTIVE
       34 A5    00000000'EF    D0  020B   381          MOVL    SCH$WAKE,PKT_WAKE(R5)
       20 A5    00000000'EF    D0  0213   382          MOVL    EXE$DEANONPAGED,PKT_DEANON(R5)
                               021B   383 ;        MOVL    EXE$MCHK_PRTCT,PKT_PRTCT(R5)
                      52   04   9A  021B   384          MOVZBL  #PRI$_TICOM,R2           ; SET PRIORITY INCREMENT CLASS
                   50    0C A5   3C  021E   385          MOVZWL  ACB$L_PID(R5),R0         ; GET DESTINATION PID
             51    00000000'FF    D0  0222   386          MOVL    @SCH$GL_PCBVEC,R1        ; GET ADDRESS OF PCB VECTOR
                      54    6140   D0  0229   387          MOVL    (R1)[R0],R4              ; GET DESTINATION PCB ADDRESS
                                  022D   388          SETIPL  #IPL$_SYNCH              ; DON'T LET TARGET'S STATE CHANGE
                   09   2C A4   B1  0230   389          CMPW    PCB$W_STATE(R4),#SCH$C_SUSP      ; IF TARGET PROCESS SUSPENDED
                            0D   13  0234   390          BEQL    100$                             ; THEN RESUME IT
                   0A   2C A4   B1  0236   391          CMPW    PCB$W_STATE(R4),#SCH$C_SUSPO     ; OR SUSPENDED AND OUTSWAPPED
                            07   13  023A   392          BEQL    100$                             ; THEN RESUME IT
                      1C B5   16  023C   393          JSB     @PKT_QAST(R5)            ; QUEUE AST FOR TARGET (NO RESUSPEND)
                                  023F   394          SETIPL  #0                       ; DROP IPL, BLOCK IS GONE
```

```
                     04  0242   395          RET                                 ; RETURN TO ORIGINAL MODE
                         0243   396 ;
                         0243   397 ;       RESUME DESTINATION PROCESS AFTER QUEUEING THE SPECIAL KERNEL AST.
                         0243   398 ;       NOTE THAT BECAUSE THIS AST WILL PREEMPT THE SUSPND AST CODE IN
                         0243   399 ;       THE EXEC, THE RESPEN BIT WILL NOT GET CLEARED PRIOR TO THIS AST'S
                         0243   400 ;       EXECUTION. THUS, THIS AST MERELY HAS TO CLEAR THE RESPEN BIT TO
                         0243   401 ;       ENSURE THAT THE TARGET PROCESS REENTERS THE SUSPENDED STATE. THE EXEC
                         0243   402 ;       WILL GET A CHANCE TO RUN WHEN THIS AST COMPLETES, IT WILL SEE THAT
                         0243   403 ;       RESPEN BIT CLEAR, AND IMMEDIATELY REENTER THE SUSPEND STATE.
                         0243   404 ;
  3C A5    01     90    0243   405 100$:   MOVB    #PKT_M_SUSPEND,PKT_FLAGS(R5)    ; MARK PROCESS IN SUSPEND STATE
        1C B5    16    0247   406          JSB     @PKT_QAST(R5)                   ; QUEUE AST FOR TARGET (RESUSPEND)
                       024A   407          SETIPL  #0                              ; LOWER IPL, BLOCK IS NOW GONE
  50    60 A4    D0    024D   408          MOVL    PCB$L_PID(R4),R0                ; GET PROCESS IPID
00000000'FF      16    0251   409          JSB     @EXE$IPID_TO_EPID               ; CONVERT TO EPID (IN R0)
           50    DD    0257   410          PUSHL   R0                              ; PUSH EPID ON STACK
     50    5E    D0    0259   411          MOVL    SP,R0                           ; POINT TO IT
                       025C   412          $RESUME_S PIDADR=(R0)                   ; RESUME PROCESS SO AST WILL EXECUTE
                     04  0267   413          RET                                 ; RETURN
```

```
                                0268   415 ;
                                0268   416 ;       CODE PLACED IN NON-PAGED BUFFER EXECUTED IN
                                0268   417 ;       DESTINATION PROCESS CONTEXT AS A SPECIAL KERNEL AST.
                                0268   418 ;
                                0268   419 CODE:   IFNORD  PKT_LEN(R5),@PKT_ADDR(R5),10$   ; BRANCH IF NOT READABLE
                          30    BB    0270   420          PUSHR   #^M<R4,R5>                ; SAVE REGISTERS
                                0272   421 ;        PUSHAB  B^5$                      ; END OF RECOVERY BLOCK ADDESS
                                0272   422 ;        MOVL    #<MCHK$M_LOG!MCHK$M_MCK!MCHK$M_NEXM!MCHK$M_UBA>,R0 ; PROTECT MASK
                                0272   423 ;        JSB     @PKT_PRTCT(R5)            ; INHIBIT MACHINE CHECKS
    3D A5    14 B5    28 A5    28   0272   424          MOVC    PKT_LEN(R5),@PKT_ADDR(R5),PKT_DATABUF(R5) ; GET DATA
                                0279   425          RSB                               ; END OF PROTECTED CODE
                          30    BA    0279   426 5$:       POPR    #^M<R4,R5>                ; RESTORE REGISTERS
                                027B   427 ;        BLBC    R0,10$                    ; BRANCH IF MACHINE CHECK OCCURRED
                       2C A5    01    D0    027B   428          MOVL    #SS$_NORMAL,PKT_STATUS(R5) ; SET SUCCESS ON TRANSFER
                                027F   429 10$:      ASSUME  PKT_V_SUSPEND EQ 0
                    05 3C A5    E9    027F   430          BLBC    PKT_FLAGS(R5),40$         ; BRANCH IF NOT RE-SUSPENDING
                00 24 A4    05    E5    0283   431          BBCC    #PCB$V_RESPEN,PCB$L_STS(R4),40$ ; ALLOW TARGET TO REMAIN IN SUSP
                0C A5    10 A5    D0    0288   432 40$:      MOVL    PKT_ORIGPID(R5),ACB$L_PID(R5)    ; SET PID FOR RETURN AST
                0B A5    80 8F    90    028D   433          MOVB    #1@ACB$V_KAST,ACB$B_RMOD(R5)     ; SET FOR KAST AGAIN
                18 A5    9D'AF    9E    0292   434          MOVAB   B^REPLY,ACB$L_KAST(R5)   ; SET NEW AST ADDRESS
                      52    04    9A    0297   435          MOVZBL  #PRI$_TICOM,R2           ; SET PRIORITY INCREMENT CLASS
                         1C B5    17    029A   436          JMP     @PKT_QAST(R5)            ; QUEUE RETURN AST AND EXIT
                                029D   437 ;
                                029D   438 ;       CODE PLACED IN NON-PAGED BUFFER EXECUTED IN
                                029D   439 ;       ORIGINATOR PROCESS CONTEXT TO RETURN MEMORY
                                029D   440 ;       TO REQUESTED BUFFER AND RETURN COMPLETION STATUS.
                                029D   441 ;
                50    6C A4    D0    029D   442 REPLY:   MOVL    PCB$L_PHD(R4),R0         ; GET ADDRESS OF PROCESS HEADER
             00F4 C0    38 A5    D1    02A1   443          CMPL    PKT_IMGCNT(R5),PHD$L_IMGCNT(R0) ; CHECK IF STILL SAME IMAGE
                           2A    12    02A7   444          BNEQ    DEALOC                   ; IF NOT, DROP TRANSFER ON FLOOR
                                02A9   445          IFNOWRT PKT_LEN(R5),@PKT_RETLOC(R5),130$ ; BRANCH IF NOT WRITABLE
                           55    DD    02B1   446          PUSHL   R5                       ; SAVE REGISTER
    24 B5    3D A5    28 A5    28   02B3   447          MOVC    PKT_LEN(R5),PKT_DATABUF(R5),@PKT_RETLOC(R5) ; MOVE DATA
                           55 8ED0    02BA   448          POPL    R5                       ; RESTORE REGISTER
                                02BD   449 130$:     IFNOWRT #4,@PKT_STATLOC(R5),140$ ; BRANCH IF STATUS NOT WRITABLE
             30 B5    2C A5    D0    02C4   450          MOVL    PKT_STATUS(R5),@PKT_STATLOC(R5) ; RETURN STATUS
                51    0C A5    D0    02C9   451 140$:     MOVL    ACB$L_PID(R5),R1         ; GET PID FOR WAKE
                                02CD   452          SETIPL  #IPL$_SYNCH              ; RAISE TO SYNCH
                      34 B5    16    02D0   453          JSB     @PKT_WAKE(R5)            ; WAKE REQUESTOR PROCESS
                                02D3   454
                                02D3   455
                                02D3   456 DEALOC:  SETIPL  #IPL$_ASTDEL             ; RESTORE IPL
                   50    55    D0    02D6   457          MOVL    R5,R0                    ; SET ADDRESS FOR RELEASE
                     20 B5    17    02D9   458          JMP     @PKT_DEANON(R5)          ; FREE BLOCK
                                02DC   459
                         00000074   02DC   460 CODELEN = .-CODE                  ; SIZE OF ENTIRE CODE SEGMENT
                                02DC   461
                                02DC   462          .END
```

```
$$T1                    = 00000001
ACB$B_RMOD              = 0000000B
ACB$L_AST               = 00000010
ACB$L_ASTPRM            = 00000014
ACB$L_KAST              = 00000018
ACB$L_PID               = 0000000C
ACB$V_KAST              = 00000007
CODE                      00000268 R      03
CODELEN                 = 00000074
DEALOC                    000002D3 R      03
EXE$ALLOCBUF              ********   X    03
EXE$DEANONPAGED           ********   X    03
EXE$GL_RPB                ********   X    03
EXE$IPID_TO_EPID          ********   X    03
GETPROCMEM                00000008 RG     03
IPL$_ASTDEL             = 00000002
IPL$_SYNCH              = 00000008
MMG$GL_MAXGPTE            ********   X    03
MMG$GL_SPTBASE            ********   X    03
PCB$L_PHD               = 0000006C
PCB$L_PID               = 00000060
PCB$L_STS               = 00000024
PCB$V_RESPEN            = 00000005
PCB$W_STATE             = 0000002C
PHD$L_IMGCNT            = 000000F4
PHD$W_PHVINDEX          = 00000042
PHV$GL_PIXBAS             ********   X    03
PKT_ADDR                = 00000014
PKT_DATABUF               0000003D
PKT_DEANON                00000020
PKT_FLAGS                 0000003C
PKT_IMGCNT                00000038
PKT_LEN                   00000028
PKT_M_SUSPEND           = 00000001
PKT_ORIGPID             = 00000010
PKT_QAST                  0000001C
PKT_RETLOC                00000024
PKT_SIZE                  0000003D
PKT_STATLOC               00000030
PKT_STATUS                0000002C
PKT_V_SUSPEND           = 00000000
PKT_WAKE                  00000034
PR$_IPL                 = 00000012
PRI$_TICOM              = 00000004
PSL$C_KERNEL            = 00000000
PSL$S_PRVMOD            = 00000002
PSL$V_PRVMOD            = 00000016
PTE$S_PFN               = 00000015
PTE$V_PFN               = 00000000
QAST                      000000B0 R      03
QAST_COUNT                00000000 R      02
QAST_TIMEOUT              0000007A R      03
REJECT_RESPONSE           0000009E R      03
REPLY                     0000029D R      03
RPB$C_MEMDSCSIZ         = 00000008
RPB$C_NMEMDSC           = 00000008
RPB$L_MEMDSC            = 000000BC
RPB$S_BASEPFN           = 00000020
RPB$S_PAGCNT            = 00000018
RPB$V_BASEPFN           = 00000020
RPB$V_PAGCNT            = 00000000
SCH$C_SUSP              = 00000009
SCH$C_SUSPO             = 0000000A
SCH$GL_CURPCB             ********   X    03
SCH$GL_MAXPIX             ********   X    03
SCH$GL_PCBVEC             ********   X    03
SCH$QAST                  ********   X    03
SCH$WAKE                  ********   X    03
SECONDS                 = 00989680
SGN$GL_BALSETCT           ********   X    03
SIZ...                  = 00000001
SS$_ACCVIO              = 0000000C
SS$_NONEXPR             = 000008E8
SS$_NORMAL              = 00000001
SS$_TIMEOUT             = 0000022C
SWP$GL_BALBASE            ********   X    03
SWP$GL_BSLOTSZ            ********   X    03
SYS$CANTIM                ********  GX    03
SYS$CMKRNL                ********  GX    03
SYS$HIBER                 ********  GX    03
SYS$RESUME                ********  GX    03
SYS$SETIMR                ********  GX    03
SYS$WAKE                  ********  GX    03
TIMEOUT                   00000000 R      03
VA$S_VPN                = 00000015
VA$V_SYSTEM             = 0000001F
VA$V_VPN                = 00000009
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	0000003D (61.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
DATA	00000004 (4.)	02 (2.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC BYTE
QAST	000002DC (732.)	03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	35	00:00:00.04	00:00:01.10
Command processing	134	00:00:00.43	00:00:03.77
Pass 1	322	00:00:07.38	00:00:26.97
Symbol table sort	0	00:00:01.14	00:00:05.55
Pass 2	103	00:00:01.48	00:00:06.06
Symbol table output	11	00:00:00.05	00:00:00.48
Psect synopsis output	2	00:00:00.02	00:00:00.36
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	609	00:00:10.54	00:00:44.29

The working set limit was 1650 pages.
60357 bytes (118 pages) of virtual memory were used to buffer the intermediate code.
There were 60 pages of symbol table space allocated to hold 1053 non-local and 19 local symbols.
462 source lines were read in Pass 1, producing 20 object records in Pass 2.
36 pages of virtual memory were used to define 35 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SDA.OBJ]SDALIB.MLB;1	0
_255DUA28:[SYS.OBJ]LIB.MLB;1	13
_255DUA28:[SYSLIB]STARLET.MLB;2	18
TOTALS (all libraries)	31

1222 GETS were required to define 31 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:QAST/OBJ=OBJ$:QAST MSRC$:QAST/UPDATE=(ENH$:QAST)+EXECML$/LIB+LIB$:SDALIB/LIB
```

PROCESS
LIS

RMS
LIS

QAST
LIS

RELEASE
LIS

POOL
LIS